The vga_sync circuit generates timing and synchronization signals. The hsync and vsync signals are connected to the VGA port to control the horizontal and vertical scans of the monitor. The two signals are decoded from the internal counters, whose outputs are the pixel_x and pixel_y signals. The pixel_x and pixel_y signals indicate the relative positions of the scans and essentially specify the location of the current pixel. The vga_sync circuit also generates the video_on signal to indicate whether to enable or disable the display. The design of this circuit is discussed in Section 13.2.

The pixel generation circuit generates the three video signals, which are collectively referred to as the rgb signal. A color value is obtained according to the current coordinates of the pixel (the pixel_x and pixel_y signals) and the external control and data signals. This circuit is more involved and is discussed in the second half of this chapter and Chapter 14.

13.2 VGA SYNCHRONIZATION

The video synchronization circuit generates the hsync signal, which specifies the required time to traverse (scan) a row, and the vsync signal, which specifies the required time to traverse (scan) the entire screen. Subsequent discussions are based on a 640-by-480 VGA screen with a 25-MHz pixel rate, which means that 25M pixels are processed in a second. Note that this resolution is also known as the VGA mode.

The screen of a CRT monitor usually includes a small black border, as shown at the top of Figure 13.4. The middle rectangle is the visible portion. Note that the coordinate of the vertical axis increases downward. The coordinates of the top-left and bottom-right corners are (0,0) and (639,479), respectively.

13.2.1 Horizontal synchronization

A detailed timing diagram of one horizontal scan is shown in Figure 13.4. A period of the hsync signal contains 800 pixels and can be divided into four regions:
Figure 13.4 Timing diagram of a horizontal scan.
- **Display**: region where the pixels are actually displayed on the screen. The length of this region is 640 pixels.
- **Retrace**: region in which the electron beams return to the left edge. The video signal should be disabled (i.e., black), and the length of this region is 96 pixels.
- **Right border**: region that forms the right border of the display region. It is also known as the **front porch** (i.e., porch before retrace). The video signal should be disabled, and the length of this region is 16 pixels.
- **Left border**: region that forms the left border of the display region. It is also known as the **back porch** (i.e., porch after retrace). The video signal should be disabled, and the length of this region is 48 pixels.

Note that the lengths of the right and left borders may vary for different brands of monitors.

The hayscale signal can be obtained by a special mod-800 counter and a decoding circuit. The counts are marked on the top of theayscale signal in Figure 13.4. We intentionally start the counting from the beginning of the display region. This allows us to use the counter output as the horizontal (x-axis) coordinate. This output constitutes the pixel.x signal. Theayscale signal goes low when the counter's output is between 656 and 751.

Note that the CRT monitor should be black in the right and left borders and during retrace. We use the v_video_on signal to indicate whether the current horizontal coordinate is in the displayable region. It is asserted only when the pixel count is smaller than 640.

### 13.2.2 Vertical synchronization

During the vertical scan, the electron beams move gradually from top to bottom and then return to the top. This corresponds to the time required to refresh the entire screen. The format of the vsync signal is similar to that of theayscale signal, as shown in Figure 13.5. The time unit of the movement is represented in terms of horizontal scan lines. A period of the vsync signal is 525 lines and can be divided into four regions:

- **Display**: region where the horizontal lines are actually displayed on the screen. The length of this region is 480 lines.
• **Retrace:** region that the electron beams return to the top of the screen. The video signal should be disabled, and the length of this region is 2 lines.

• **Bottom border:** region that forms the bottom border of the display region. It is also known as the front porch (i.e., porch before retrace). The video signal should be disabled, and the length of this region is 10 lines.

• **Top border:** region that forms the top border of the display region. It is also known as the back porch (i.e., porch after retrace). The video signal should be disabled, and the length of this region is 33 lines.

As in the horizontal scan, the lengths of the top and bottom borders may vary for different brands of monitors.

The *v sync* signal can be obtained by a special mod-525 counter and a decoding circuit. Again, we intentionally start counting from the beginning of the display region. This allows us to use the counter output as the vertical (y-axis) coordinate. This output constitutes the *pixel y* signal. The *v sync* signal goes low when the line count is 490 or 491.

As in the horizontal scan, we use the *v video on* signal to indicate whether the current vertical coordinate is in the displayable region. It is asserted only when the line count is smaller than 480.

### 13.2.3 Timing calculation of VGA synchronization signals

As mentioned earlier, we assume that the pixel rate is 25 MHz. It is determined by three parameters:

• **p:** the number of pixels in a horizontal scan line. For 640-by-480 resolution, it is

\[
p = 800 \frac{\text{pixels}}{\text{line}}
\]

• **l:** the number of lines in a screen (i.e., a vertical scan). For 640-by-480 resolution, it is

\[
l = 525 \frac{\text{lines}}{\text{screen}}
\]

• **s:** the number of screens per second. For flickering-free operation, we can set it to

\[
s = 60 \frac{\text{screens}}{\text{second}}
\]

The *s* parameter specifies how fast the screen should be refreshed. For a human eye, the refresh rate must be at least 30 screens per second to make the motion appear to be continuous. To reduce flickering, the monitor usually has a much higher rate, such as the 60 screens per second specification above. The pixel rate can be calculated by the three parameters:

\[
\text{pixel rate} = p \times l \times s \approx 25M \frac{\text{pixels}}{\text{second}}
\]

The pixel rate for other resolutions and refresh rates can be calculated in a similar fashion. Clearly, the rate increases as the resolution and refresh rate grow.

### 13.2.4 HDL implementation

The function of the *vga sync* circuit is discussed in Section 13.1.3. If the frequency of the system clock is 25 MHz, the circuit can be implemented by two special counters: a
mod-800 counter to keep track of the horizontal scan and a mod-525 counter to keep track of the vertical scan.

Since our designs generally use the 50-MHz oscillator of the prototyping board, the system clock rate is twice the pixel rate. Instead of creating a separate 25-MHz clock domain and violating the synchronous design methodology, we can generate a 25-MHz enable tick to enable or pause the counting. The tick is also routed to the p_tick port as an output signal to coordinate operation of the pixel generation circuit.

The HDL code is shown in Listing 13.1. It consists of a mod-2 counter to generate the 25-MHz enable tick and two counters for the horizontal and vertical scans. We use two status signals, h_end and v_end, to indicate completion of the horizontal and vertical scans. The values of various regions of the horizontal and vertical scans are defined as constants. They can easily be modified if a different resolution or refresh rate is used. To remove potential glitches, output buffers are inserted for the hsync and vsync signals. This leads to a one-clock-cycle delay. We should add a similar buffer for the rgb signal in the pixel generation circuit to compensate for the delay.

Listing 13.1 VGA synchronization circuit

```vhdl
module vga_sync

begin

    // constant declaration
    // VGA 640-by-480 sync parameters
    localparam HD = 640;   // horizontal display area
    localparam HF = 48;    // h. front (left) border
    localparam HB = 16;    // h. back (right) border
    localparam HR = 96;    // h. retrace
    localparam VD = 480;   // vertical display area
    localparam VF = 10;    // v. front (top) border
    localparam VB = 33;    // v. back (bottom) border
    localparam VR = 2;     // v. retrace

    // mod-2 counter
    reg mod2_reg;
    wire mod2_next;
    // sync counters
    reg [9:0] h_count_reg, h_count_next;
    reg [9:0] v_count_reg, v_count_next;
    // output buffer
    reg v_sync_reg, h_sync_reg;
    wire v_sync_next, h_sync_next;
    // status signal
    wire h_end, v_end, pixel_tick;

    // body
    // registers
    always @(posedge clk, posedge reset)
        if (reset)
            begin
```
mod2_reg <= 1'b0;
v_count_reg <= 0;
h_count_reg <= 0;
v_sync_reg <= 1'b0;
h_sync_reg <= 1'b0;
end
else
begin
    mod2_reg <= mod2_next;
v_count_reg <= v_count_next;
h_count_reg <= h_count_next;
v_sync_reg <= v_sync_next;
h_sync_reg <= h_sync_next;
end

// mod-2 circuit to generate 25 MHz enable tick
assign mod2_next = ~mod2_reg;
assign pixel_tick = mod2_reg;

// status signals
// end of horizontal counter (799)
assign h_end = (h_count_reg==((HD+HF+HB+HR-1)));
// end of vertical counter (524)
assign v_end = (v_count_reg==((VD+VF+VB+VR-1)));

// next-state logic of mod-800 horizontal sync counter
always @*
    if (pixel_tick) // 25 MHz pulse
        if (h_end)
            h_count_next = 0;
        else
            h_count_next = h_count_reg + 1;
    else
        h_count_next = h_count_reg;

// next-state logic of mod-525 vertical sync counter
always @*
    if (pixel_tick & h_end)
        if (v_end)
            v_count_next = 0;
        else
            v_count_next = v_count_reg + 1;
    else
        v_count_next = v_count_reg;

// horizontal and vertical sync, buffered to avoid glitch
// h_sync_next asserted between 636 and 751
assign h_sync_next = (h_count_reg>=(HD+HB)) & &
        h_count_reg<=(HD+HB+HR-1));
// vh_sync_next asserted between 490 and 491
assign v_sync_next = (v_count_reg>=(VD+VB)) & &
        v_count_reg<=(VD+VB+VR-1));
// video on/off
assign video_on = (h_count_reg<HD) && (v_count_reg<VD);

// output
assign hsync = h_sync_reg;
assign vsync = v_sync_reg;
assign pixel_x = h_count_reg;
assign pixel_y = v_count_reg;
assign p_tick = pixel_tick;
endmodule

13.2.5 Testing circuit

To verify operation of the synchronization circuit, we can connect the rgb signal to three switches. The entire visible region should be turned on with a single color. We can go through the eight possible combinations and check the colors defined in Table 13.1. The HDL code is shown in Listing 13.2. As mentioned in Section 13.2.4, an output buffer is added for the rgb signal.

<table>
<thead>
<tr>
<th>Listing 13.2</th>
<th>VGA synchronization testing circuit</th>
</tr>
</thead>
<tbody>
<tr>
<td>module vga_test</td>
<td></td>
</tr>
<tr>
<td>(</td>
<td></td>
</tr>
<tr>
<td>input wire clk, reset,</td>
<td></td>
</tr>
<tr>
<td>input wire [2:0] sw,</td>
<td></td>
</tr>
<tr>
<td>output wire hsync, vsync,</td>
<td></td>
</tr>
<tr>
<td>output wire [2:0] rgb</td>
<td></td>
</tr>
<tr>
<td>);</td>
<td></td>
</tr>
<tr>
<td>//signal declaration</td>
<td></td>
</tr>
<tr>
<td>reg [2:0] rgb_reg;</td>
<td></td>
</tr>
<tr>
<td>wire video_on;</td>
<td></td>
</tr>
<tr>
<td>// instantiate vga sync circuit</td>
<td></td>
</tr>
<tr>
<td>vga_sync vsync_unit</td>
<td></td>
</tr>
<tr>
<td>(.clk(clk), .reset(reset), .hsync(hsync), .vsync(vsync),</td>
<td></td>
</tr>
<tr>
<td>.video_on(video_on), .p_tick(), .pixel_x(), .pixel_y());</td>
<td></td>
</tr>
<tr>
<td>// rgb buffer</td>
<td></td>
</tr>
<tr>
<td>always @(posedge clk, posedge reset)</td>
<td></td>
</tr>
<tr>
<td>if (reset)</td>
<td></td>
</tr>
<tr>
<td>rgb_reg &lt;= 0;</td>
<td></td>
</tr>
<tr>
<td>else</td>
<td></td>
</tr>
<tr>
<td>rgb_reg &lt;= sw;</td>
<td></td>
</tr>
<tr>
<td>// output</td>
<td></td>
</tr>
<tr>
<td>assign rgb = (video_on) ? rgb_reg : 3'b0;</td>
<td></td>
</tr>
<tr>
<td>endmodule</td>
<td></td>
</tr>
</tbody>
</table>